**ABSTRACT**

In my Neural Networks and Intelligent Machine project I want to train the machine to predict housing prices. The project will examine how the machine will use neural networks to examine previous data on housing prices based off a list of attributes that affect housing prices. Using Columns and Rows to relate the data, coding techniques, and algorithms to train the neural network to compute loss, accuracy, and regularization. This will show a set of graphs to help visualize the machines computations. The trend of housing prices based on the attributes of the house which are estimated in the results.

**GENERAL TERMS**

Some terms to understand for this paper:

Regularization: Reduces error from a model by avoiding overfitting and training the model to function properly.

Overfitting: Is when you overfeed the model with data that does not contain the capacity to handle it.

Python: A high-level, general purpose, interpreted object-oriented programming language.

Neural Networks: A network of artificial neurons programmed in software.

Packages: A complete documented set of programs designed to be supplied to several users for a single application or function.

**INTRODUCTION**

Setting up the environment to process the data and machine learning I used Anaconda an open-source package and environment management system. Anaconda makes it easier to install and update packages as well as create and load environments. I was able to use Jupyter Notebook with Anaconda an open-source web application to create and share computational documents. Jupyter Notebook allows us to run python code on the web browser in snippets without having to run the whole program. The data of house attributes that relates to housing prices are imported into the Jupyter Notebook so that the Anaconda can access the data. The data is set into an array that works with a formula to graph the analysis. The dataset is split into input features and a label. Then finally, the dataset is scaled and split into training, validation, and test set. Once the data is set to evaluate, I used Anaconda to visualize the accuracy and loss of the housing price predictions by recognizing overfitting using regularization to reduce overfitting. By the end of the project, I used the Test set to give us an accurate reporting of housing price prediction.

**What are Prediction techniques?**

Prediction techniques used in this deep-learning project are popular techniques used in many other set ups for neural networks. The basis of neural networks functions off many layers like the neurons in our brain. Layers receive input such as images, videos, sound, text and etc. Neural networks are a network of artificial neurons programmed in software. The input of one layer is fed into the next layer. The networks are very complex and consist of many parameters that classify and recognize the input received.[[1]](#footnote-1) The prediction process of neural networks uses a computing system of interconnected nodes. The system uses a cluster of large and raw data sets which finds patterns from which it can solve complex problems and make intelligent predictions calculating a range of accuracy. The packages used in this project provide the functionality for processing the data set in neural network.

*TensorFlow and Keras*

These packages are helpful for us to plug-and-play code for deep learning. A tensor is a mathematical object in some n-dimensional vector space. The flow graph is another mathematical object. The graph is a directed graph in this case with TensorFlow the tensor data flows through the graph as edges while being operated on at the nodes. Keras is written in python that works with scikit-learn a later package used. Keras is also used in the backend of Theano or Tensorflow as a high-level interface. The core component of Keras architecture is a model with layers, activations, optimization, and loss.[[2]](#footnote-2)

*Pandas, Scikit-learn, and matplotlib*

These packages are common for data scientist to process data as well as to visualize graphs inside of Jupyter Notebook. Pandas is a python library for data manipulation and analysis. Pandas works with matplotlib and scikit-learn later in the code. Scikit-learn is a robust machine learning library that has a vast number of algorithms ready for use in machine leaning models. Matplotlib is a plotting library for python using NumPy handling numerical data. Specifically, matplotlib has to do with visual plotting tools.

**Processing the Data**

To begin importing the packages to Jupyter Notebook is required to explore processing the data.

**Graphical user interface, text, application, email

Description automatically generated**

The Data for all the housing attributes needs to be imported into Jupyter Notebook using the following code.



This is a list of the attributes that affect the housing prices used as inputs:

* Lot Area (in sq ft)
* Overall Quality (scale from 1 to 10)
* Overall Condition (scale from 1 to 10)
* Total Basement Area (in sq ft)
* Number of Full Bathrooms
* Number of Half Bathrooms
* Number of Bedrooms above ground
* Total Number of Rooms above ground
* Number of Fireplaces
* Garage Area (in sq ft)

To set the data into an array we use dataset a function from matplotlib.

Diagram

Description automatically generated

The Data needs to be split into the input and output of what to predict. The goal is to predict if the housing price is over or below the median using 1 for yes and 0 for no. one part for preprocessing is scaling the data once it is split into X,Y value. X is used for the first 10 columns 0-9 and Y is used for the last column which is the prediction column. Scikit-learn is imported to scale the data.

Graphical user interface, text, application, email

Description automatically generated

The dataset is split into 3 sets of one training, one validation, and a test set. The scikit-learn package calls ‘train\_test\_split to split the dataset using the following code to do so.

Graphical user interface, text, application, email

Description automatically generated

After splitting, we have six variables to use for the datasets listed below.

* X\_train (10 input features, 70% of full dataset)
* X\_val (10 input features, 15% of full dataset)
* X\_test (10 input features, 15% of full dataset)
* Y\_train (1 label, 70% of full dataset)
* Y\_val (1 label, 15% of full dataset)
* Y\_test (1 label, 15% of full dataset)

**Setting up the neural network**

The neural network architecture we want to use is a sequential model in Keras which looks like this model below.[[3]](#footnote-3)

Diagram, schematic

Description automatically generated

These layers are identified by:

* Hidden layer 1: 32 neurons, ReLU activation
* Hidden layer 2: 32 neurons, ReLU activation
* Output Layer: 1 neuron, Sigmoid activation

The following code is used to initialize the architecture and describe the layers to Keras. Dense refers to a fully connected layer which is used.

Graphical user interface, text, application

Description automatically generated

The model is required to find the best numbers to use for optimizing, loss, and accuracy. The code model.compile is used to tell the program to use stochastic gradient descent (sgd) as an optimizer, and binary cross entropy for the loss function. Finally, metrics is used to track the accuracy on the loss function.

Text

Description automatically generated

At this point the neural network is ready to train by implementing model.fit to fit the parameters of the data. The model specifies what to train, how much to train, and for how long. The following code shows this training process and saves it under the variable hist which is saved for later visualization in Analysis.

Text

Description automatically generated

Final code in finding the accuracy of test sets predictions is model.evaluate function which returns loss as the first element 0 and accuracy as the second element 1.



**Data Analysis and Results**

In the analysis of the data, it is important to check whether the model is affected by overfitting. Graphs are used to check these models. The graphs are training loss compared to validation loss and training accuracy compared to validation accuracy. The following code is for Training loss and Validation loss once matplotlib is imported.



Text

Description automatically generated with medium confidence

The next code is used for training accuracy and validation accuracy.

Text

Description automatically generated

From both graphs produced shown below it shows that the model doesn’t have a problem with overfitting.

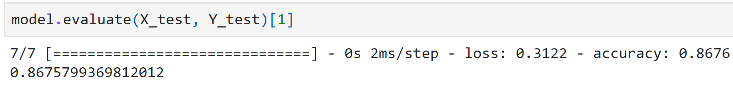
Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

The analysis of the data shows that the accuracy of the housing prices prediction model is 80-90% accurate. Showing the range, we get when evaluating our test set.



**Discussion**

In developing the model, I noticed that depending on different attributes for housing prices used in the model could make a difference in predictions. Although most attributes being listed gives the housing prediction model a well-balanced model for accurate predictions. The use of neural networks for predicting housing prices will be a valuable tool for future data analysis for buyers, sellers, and anyone else interested in real estate. Understanding the importance of neural networks in this aspect has taught me a lot of possibilities that deep-learning tools could do for the future. I also was able to learn how to deal with over-fitting in prediction models and how Regularization could be implemented in cases like that using Keras.

**Conclusion**

In conclusion the model the machine learning model proved to work with good accuracy. Anaconda and Jupyter Notebook were good for implementing the right packages to do the job using neural networking tools. Prediction by neural networks training algorithms is a growing field in computer science and are proving to be successful. This project is an example of growing use neural networks and shows the accuracy for a prediction model based on neural networks. The historical data saved on housing prices will benefit the future prediction models. Comparing these predictions over time will show which direction the housing prices are going based on previous datasets. Whether the housing prices are above or below the median price. In the prediction model the calculated loss turned out be around 30% within the period of study with an accuracy of 80-90% meaning that our model is performing well determining housing price trend. Training this neural network to perform similar tasks in the future will be beneficial for quick pricing analysis on houses.

**Reference**

[1] Francois Chollet, Deep learning with Python, 2018

[2] Dr Donald Kinghorn, TensorFlow Introduction What is Tensorflow, 2018

[3] Joseph Lee Wei En, Building your first neural network to predict house prices with Keras, 2019

1. Francois Chollet, 2018 [↑](#footnote-ref-1)
2. Dr Donald Kinghorn, 2018 [↑](#footnote-ref-2)
3. Joseph Lee Wei En,2019 [↑](#footnote-ref-3)